

PRACTICAL: 3

AIM:

Languages exhibit distinct statistical characteristics, with certain letters appearing more frequently than others. For example, in English, the letter 'e' is the most common, followed by 't', 'a', and others. By comparing the frequency of characters in the encrypted text to this standard distribution, we can hypothesize which symbols correspond to which letters in the original language. Implement how statistical patterns in a language can assist in decoding an encrypted message. The encrypted text has been generated using a consistent letter-shifting method. Our objective is to analyze the frequency of characters within the text and uncover the underlying pattern to reconstruct the original message.

THEORY:

Introduction to Frequency Analysis

Frequency analysis is a classical cryptographic technique used to break ciphers that involve substitution, such as the Caesar cipher or monoalphabetic substitution cipher. The core principle of frequency analysis relies on the fact that in natural languages, certain letters and patterns appear with a higher frequency than others. In the English language, for example, the letter 'e' is the most frequently occurring letter, followed by 't', 'a', and others. By analyzing the frequency of characters in an encrypted message (ciphertext), we can compare them to the typical letter distribution of the language and hypothesize which letters correspond to which in the original (plaintext) message.

The Concept of Substitution Ciphers

In a substitution cipher, each letter in the plaintext is replaced by a different letter or symbol. The most basic substitution cipher is the Caesar cipher, where each letter is shifted by a fixed number in the alphabet. For example, with a shift of 3, A becomes D, B becomes E, and so on.

Substitution ciphers, especially simple ones like the Caesar cipher, are vulnerable to frequency analysis because the frequency of letters in the ciphertext follows the same distribution as in the plaintext. If we can identify patterns and match the most frequent letters in the ciphertext to the most frequent letters in the language, we can begin to reverse the encryption process.

Why Frequency Analysis Works

Frequency analysis works because of the statistical properties of language. In English, certain letters, digraphs (pairs of letters), and trigraphs (three-letter combinations) appear more often than others. By leveraging these frequencies, we can gain insight into the structure of the encrypted message and begin to unravel the encryption.

Even in cases where the cipher uses a more complex substitution scheme, like the Vigenère cipher, frequency analysis can still be helpful, although it requires additional steps and may not work as directly as with simpler ciphers.

Encryption Formula

To encrypt a letter in the Caesar cipher, we use the following formula:

$$C=(P+k) \bmod 26$$

Where:

C = Ciphertext letter (encrypted letter)

P = Plaintext letter (original letter in the message)

k = Shift key (the number of positions each letter is shifted)

mod 26 = Ensures the result wraps around the alphabet (since there are 26 letters in the English alphabet)

Decryption Formula

To decrypt a letter, we reverse the shift (i.e., subtract the key instead of adding it):

$$P=(C-k) \bmod 26$$

Where:

P = Plaintext letter (decrypted letter)

C = Ciphertext letter (encrypted letter)

k = Shift key (the same key used during encryption)

mod 26 = Ensures the result wraps around the alphabet

CODE:

```
import java.util.Scanner;

public class crns {

    public static String decryptCaesar(String text, int shift) {
        StringBuilder decrypted = new StringBuilder();

        for (char c : text.toCharArray()) {
            if (Character.isAlphabetic(c)) {
                char base = Character.isUpperCase(c) ? 'A' : 'a';
                decrypted.append((char) (((c - base - shift + 26) % 26) + base));
            } else {
                decrypted.append(c);
            }
        }

        return decrypted.toString();
    }

    public static char findMostFrequentLetter(String text) {
        int[] frequency = new int[26];
```

```
        for (char c : text.toCharArray()) {
            if (Character.isAlphabetic(c)) {
                char upperChar = Character.toUpperCase(c);
                frequency[upperChar - 'A']++;
            }
        }

        int maxFreq = 0;
        char mostFrequent = ' ';
        for (int i = 0; i < 26; i++) {
            if (frequency[i] > maxFreq) {
                maxFreq = frequency[i];
                mostFrequent = (char) ('A' + i);
            }
        }

        return mostFrequent;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("\nEnter the encrypted text: ");
        String encryptedText = scanner.nextLine();

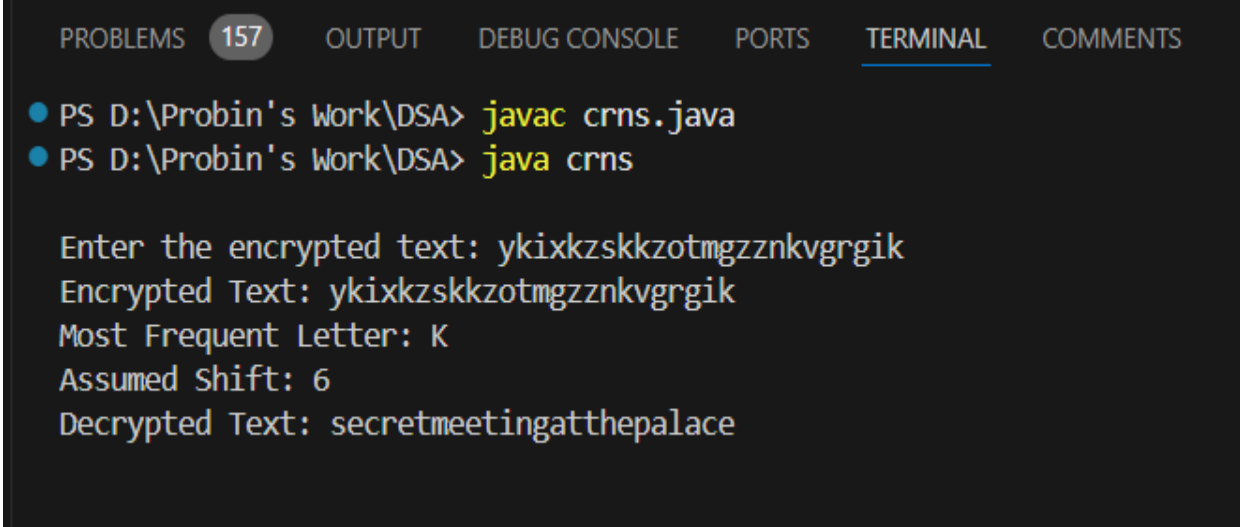
        System.out.println("Encrypted Text: " + encryptedText);

        char mostFrequent = findMostFrequentLetter(encryptedText);
        System.out.println("Most Frequent Letter: " + mostFrequent);

        int shift = (mostFrequent - 'E' + 26) % 26;
        System.out.println("Assumed Shift: " + shift);

        String decryptedText = decryptCaesar(encryptedText, shift);
        System.out.println("Decrypted Text: " + decryptedText);
        System.out.println("\n\n");

        scanner.close();
    }
}
```

OUTPUT:

```
PROBLEMS 157 OUTPUT DEBUG CONSOLE PORTS TERMINAL COMMENTS

● PS D:\Probin's Work\DSA> javac crns.java
● PS D:\Probin's Work\DSA> java crns

Enter the encrypted text: ykixkzskkzotmgzznkvgrgik
Encrypted Text: ykixkzskkzotmgzznkvgrgik
Most Frequent Letter: K
Assumed Shift: 6
Decrypted Text: secretmeetingatthepalace
```

LATEST APPLICATIONS:**1. Cryptography and Cybersecurity**

Password Strength Analysis: Frequency analysis helps evaluate password strength by identifying common patterns and character combinations used in weak passwords. It assists in creating better hashing algorithms to defend against brute-force attacks.

Breaking Weak Encryption: Older encryption methods, like substitution ciphers, can still be broken using frequency analysis by exploiting predictable letter distributions.

2. Data Compression

Huffman Coding: Frequency analysis is central to Huffman coding, a data compression technique that assigns shorter binary codes to frequently occurring characters, reducing file sizes.

Run-Length Encoding: This technique uses frequency analysis to efficiently compress data by encoding repeated patterns, like consecutive identical characters in an image.

3. Natural Language Processing (NLP)

Language Identification: Frequency analysis helps identify the language of a given text by comparing its letter frequency distribution with known language profiles.

Spell Checkers and Autocorrect: Statistical patterns of language are applied to detect and correct typographical errors based on common letter combinations.

4. Digital Forensics

Decoding Hidden Messages: Frequency analysis helps investigators decrypt messages hidden in digital media or communication, particularly in cases of steganography.

Recovering Deleted or Damaged Ciphertext: It assists in recovering compromised or deleted encrypted messages by analyzing partial ciphertext data.

LEARNING OUTCOME:

1. **Understanding Cryptography:** I learned how classical ciphers like the Caesar cipher work and how they can be broken using statistical analysis.
2. **Hands-on Cryptanalysis:** The practical helped me understand how cryptanalysis techniques, such as frequency analysis, are applied in real-world cryptographic attacks.
3. **Critical Thinking:** The practical helped me develop problem-solving skills by analyzing and decrypting a cipher systematically.

REFERENCES:

1. Tutorialspoint: <https://www.tutorialspoint.com/cryptography/index.htm>
2. GeeksforGeeks: <https://www.geeksforgeeks.org/caesar-cipher-in-cryptography>
3. Cryptography: <https://www.khanacademy.org/computing/computer-science/cryptography>